Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A method of creating run time executable code for a processing element array, comprising:

partitioning ~~a~~ the processing element array into a plurality of hardware accelerators;

decomposing a program source code into a plurality of kernel sections;

mapping said plurality of kernel sections into a plurality of hardware dependent executable code for execution on the plurality of hardware accelerators; and

forming a matrix describing different combinations of said plurality of hardware accelerators ~~and~~ , said hardware dependent executable code, and variants to support run time execution of the plurality of kernel sections by the processing element array.

2. (Original) The method of claim 1, wherein said partitioning includes partitioning into digital signal processors.

3. (Original) The method of claim 1, wherein said partitioning includes partitioning into bins.

4. (Original) The method of claim 1, wherein said mapping includes mapping into multiple hardware

5. (Original) The method of claim 4, wherein said mapping into multiple hardware contexts includes mapping a first set of variants.

6. (Original) The method of claim 5, wherein said first set of variants are produced based upon resource usage.

7. (Original) The method of claim 5, wherein said mapping includes mapping a second set of variants of said designs configured to support multiple hardware configurations of one of a plurality of bins.

8. (Original) The method of claim 1, wherein said mapping is performed by a place and route.

9. (Original) The method of claim 1, wherein said decomposing is performed manually.

10. (Original) The method of claim 1, wherein said decomposing is performed by a software profiler.

11. (Original) The method of claim 10, wherein said decomposing includes executing code compiled from said program description and monitoring timing of said executing.

12. (Original) The method of claim 11, wherein said executing utilizes a set of test data.

13. (Original) The method of claim 11, wherein said monitoring includes determining functions that consume a significant portion of said timing of said executing.

14. (Original) The method of claim 10, wherein said decomposing includes identifying kernel sections by identifying regular structures.

15. (Original) The method of claim 10, wherein said decomposing includes identifying kernel sections by identifying sections with a limited number of inputs and outputs.

16. (Original) The method of claim 10, wherein said decomposing includes identifying kernel sections by identifying sections with a limited number of branches.

17. (Original) The method of claim 10, wherein decomposing identifies overhead sections.

18. (Original) The method of claim 1, wherein mapping includes creating microcode.

19. (Original) The method of claim 1, wherein said mapping includes creating context dependent configurations.

20. (Original) The method of claim 1, wherein said matrix is sparsely-populated.

21. (Original) The method of claim 1, wherein said matrix is fully-populated

22. (Currently Amended) A system for creating run time executable code for execution on a processing element array, comprising:

a plurality of hardware accelerators partitioned from ~~a~~ the processing element array;

a plurality of kernel sections created from a program source code for execution on said plurality of hardware accelerators;

a plurality of hardware dependent executable code derived from said kernel sections for execution on said plurality of hardware accelerators; and

a matrix describing different combinations of said hardware accelerators ~~and~~ , said hardware dependent executable code, and variants, configured to support run time execution on the processing element array.

23. (Original) The system of claim 22, wherein said hardware accelerators includes digital signal processors.

24. (Original) The system of claim 22, wherein said hardware accelerators includes bins.

25. (Original) The system of claim 24, wherein said bins support multiple hardware contexts.

26. (Original) The system of claim 25, wherein said bins support a first set of variants configured to support said multiple hardware contexts.

27. (Original) The system of claim 26, wherein said first set of variants are produced based upon resource usage.

28. (Original) The system of claim 27, wherein a second set of variants of said designs are configured to support multiple hardware configurations of one of said plurality of bins.

29. (Original) The system of claim 22, wherein said mapping is performed by a place and route.

30. (Original) The system of claim 22, wherein said decomposing is performed manually.

31. (Original) The system of claim 22, wherein said decomposing is performed by a software profiler.

32. (Original) The system of claim 31, wherein said software profiler executes code compiled from said program description, and monitors time consumed.

33. (Original) The system of claim 32, wherein said software profiler includes a set of test data.

34. (Original) The system of claim 32, wherein said software profiler determines functions that consume a significant portion of said time consumed.

35. (Original) The system of claim 31, wherein said software profiler is configured to identify kernel sections by identifying regular structures.

36. (Original) The system of claim 31, wherein said software profiler is configured to identify kernel sections by identifying sections with a limited number of inputs and outputs.

37. (Original) The system of claim 31, wherein said software profiler is configured to identify kernel sections by identifying sections with a limited number of branches.

38. (Original) The system of claim 31, wherein said profiler identifies overhead sections.

39. (Original) The system of claim 22, wherein said designs include microcode.

40. (Original) The system of claim 39, wherein said microcode includes context dependent configurations.

41. (Original) The system of claim 22, wherein said matrix is sparsely-populated.

42. (Original) The system of claim 22, wherein said matrix is fully-populated.

43. (Currently Amended) A machine-readable medium having stored thereon instructions for execution by a processing element~~s~~ array, which when executed by said processing element~~s~~ array perform the following:

partitioning ~~a~~ the processing element array into a plurality of hardware accelerators;

decomposing a program source code into a plurality of kernel sections;

mapping said plurality of kernel sections into a plurality of hardware dependent executable code for execution on the plurality of hardware accelerators; and

forming a matrix describing different combinations of said plurality of hardware accelerators ~~and~~ , said hardware dependent executable code, and variants to support run time execution of the plurality of kernel sections by the processing element array.

44. (Currently Amended) A system configured to create run time executable code for execution by a processing element array, comprising:

means for partitioning ~~a~~ the processing element array into a plurality of hardware accelerators;

means for decomposing a program source code into a plurality of kernel sections for execution on said plurality of hardware accelerators;

means for mapping said plurality of kernel sections into a plurality of hardware dependent executable code for execution on the plurality of hardware accelerators; and

means for forming a matrix describing different combinations of said plurality of hardware accelerators ~~and~~ , said hardware dependent executable code, and variants to support run time execution of the plurality of kernel sections by the processing element array.